DECISION SUPPORT FOR WIRELESS EVNIRONMENTS WITH APPLICATIONS TO RADAR SYSTEMS

A Thesis
Presented to
The Academic Faculty

by

David Wonderley

In Partial Fulfillment
of the Requirements for the Degree
MSECE in the
School of Electrical and Computer Engineering

Georgia Institute of Technology
August, 2017

DECISION SUPPORT FOR WIRELESS ENVIRONMENTS WITH APPLICATIONS TO RADAR SYSTEMS

Approved by:

Dr. William L. Melvin, Co-Advisor
School of Electrical and Computer Engineering
Georgia Tech Research Institute
Georgia Institute of Technology

Dr. Douglas B. Williams, Co-Advisor
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Aaron D. Lanterman
School of Electrical & Computer Engineering
Georgia Institute of Technology

Dr. Jonathan L. Odom
School of Electrical & Computer Engineering
Georgia Tech Research Institute
Georgia Institute of Technology

Date Approved: May 13, 2017

ACKNOWLEDGEMENTS

First, I am extremely grateful to Dr. Teresa Selee, Dr. Jeremy Reed, and Dr. Jonathan Odom. My time working at Georgia Tech Research Institute was greatly enhanced by their direction, and I could never have finished this journey without their incredible guidance and support.

I am also grateful to my advisors, Dr. Doug Williams and Dr. Bill Melvin, for their valuable advice, insight, and assistance over the course of this project. In particular, I would like to thank Dr. Williams for frequently making time to help refine my ideas and writing, and I would like to thank Dr. Melvin for his advice on how to improve as a presenter and as an engineer. I would also like to think Dr. Melvin for this wonderful opportunity to work on a fascinating project alongside some of the brightest minds in the field.

I would like to thank the professors at Georgia Tech for their effort and instruction as well as my coworkers at GTRI for their friendship.

Finally, to my friends and family, thank you for your continuous support throughout this entire process.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ABBREVIATIONS

EW	Electronic Warfare
EA	Electronic Attack
EP	Electronic Protection
BN	Barrage Noise
RSN	Responsive Spot Noise
DN	Doppler Noise
RFT	Range False Targets
VGPO	Velocity Gate Pull Off
HMM	Hidden Markov Model
SVM	Support Vector Machine
SINR	Signal to Interference Plus Noise Ratio
DOS	Denial of Service
WSN	Wireless Sensor Network
SNR	Signal to Noise Ratio
JSR	Jamming to Signal Ratio
GMM	Gaussian Mixture Models

SUMMARY

This thesis introduces a framework for providing decision support for systems in wireless environments. The decision support framework utilizes game-theoretic principles to produce a metric, known as utility, that quantifies the expected benefit of executing an action for given environmental variables. From a set of utility values, a set of corresponding actions can be ranked from most to least advisable, providing an objective means to determine the optimal action under the given constraints.

In the first section, the decision support framework and its major components are introduced and their purpose explained. In the second section, an adapted version of the framework is applied to the problem of electronic warfare (EW). This section highlights the design choices that were made to implement the framework so that, under various simplifying assumptions, decision support could be provided to a radar jammer. The third section introduces a simulated non-cooperative game between two players – an emitter and a jammer – in which each player attempts to maximize its own utility. Each player makes decisions using a MATLAB® implementation of the decision support framework as it was adapted for electronic warfare. The game is simulated for multiple initial conditions with the actions of the players being recorded, and those results are used to show that the players behave in a rational manner. This work concludes with an analysis of the results obtained in the simulations and a brief discussion of how future work could improve the framework's design.

CHAPTER 1

INTRODUCTION

1.1 General Overview

While it is possible for human operators to make decisions for a wireless system, automation of the decision-making process is often desirable in real-time scenarios. For example, a network router may need to decide how to prioritize servicing traffic from different clients or a radar may need to decide how to schedule its scanning pattern. In these environments, automatic decision support could dramatically decrease response time while simultaneously improving the quality of the decisions.

Fundamentally, these two examples are the same when viewed through a game-theoretic framework: the system seeks to maximize its performance subject to some restrictions. Under a game-theoretic framework, the decision-making process hinges on how the various options are evaluated. For the example of a network router, the performance could be measured by the importance of the request, and the restriction could be a finite bandwidth. A game-theoretic framework is also robust to uncertainty, which, for the example of a router, could take the form of packet loss over an unreliable connection. Such a framework would enable the router to decide which clients receive service in a utilitarian manner.

A general-purpose, game-theoretic, modular framework that has been designed to provide automatic decision support for wireless environments will be introduced in this

work. The chief design goal for this framework was flexibility, and a modular approach allows the framework to be applied to many different types of wireless applications.

This flexibility will be illustrated through an example problem that consists of a simulated competition between two agents in a wireless environment. An agent is defined as any entity that is capable of perceiving its environment and reacting to it in a rational manner [1]. In the example application, each autonomous agent is tasked with making decisions that are rational in the context of an EW environment. A solution to this problem is proposed in terms of an implementation of the decision framework, and a non-cooperative game was designed to demonstrate the effectiveness of the implementation. This game consists of two agents, a jammer and an emitter, that utilize the decision framework to choose the responses that maximize the expected utility for the time step immediately following their respective actions. For the sake of clarity, the word emitter is used to represent a radar system interested in self-protection. This game was simulated for multiple time steps and initial conditions, and the results are analyzed in Chapter 4.

The general framework consists of three modules, each of which can be altered or substituted while still maintaining overall functionality. The three modules are the adversary identification module, the utility calculator module, and the optimization module. Each of these modules draws upon a repository of the current body of knowledge, collectively known as the knowledge base. Relevant information may include observed data, environmental models, or models of known agents. The adversary identification module analyzes the information provided by the knowledge base to determine if there are any nearby competing agents. Ultimately, the adversary identification module produces a joint probability matrix called the classification matrix. This matrix is later used by the

utility calculator to compute the expected utility of each action available to the agent. Finally, the optimization module sorts the actions according to their utility, in a manner determined by user preference.

This work describes in more detail the material presented at the 2016 IEEE Radar Conference [2].

1.2 Utility and Utility Functions

In order for an agent to make rational decisions, it must be able to predict the effectiveness of its potential actions. The expected benefit an agent receives from an action is called the utility of the action. Individual utility values are calculated from utility functions that map one or more parameters of interest to the unit interval.

Utility functions allow dissimilar actions to be directly compared. For example, in a wireless system, if utility represents signal strength, then a utility function could evaluate utility as a function of the distance between a transmitter and the receiver, where the utility decreases as the distance increases. For the same application, a separate utility function could evaluate the utility as a function of the transmitted power. The resulting utility values and, by proxy, the parameters they represent could then be directly compared against each other. While distance, transmitted power, and signal strength are all physically related, utility functions also allow comparisons between parameters that are not so easily relatable, such as transmitted power and the modulation scheme.

Additionally, utility functions provide a way for an agent to make rational decisions in the presence of uncertainty. In [3], von Neumann and Morgenstern laid out the foundations for game theory and utility-based decision making. They argued that a rational agent acts to maximize its expected utility and showed that, as long as certain axioms are

satisfied, a utility function can be formulated that allows the agent to behave rationally. An agent whose actions are chosen through a method satisfying the axioms of completeness, transitivity, continuity, and independence as expressed in [3] is said to be von Neumann-Morgenstern (VNM) rational. Throughout this work, the utility of an action will be represented by a single-parameter utility function that satisfies VNM rationality. VNM-rational utility functions allow an agent to compare an arbitrary number of complex actions simultaneously, even in the presence of uncertainty. However, in general, finding a meaningful mapping for each action while ensuring that VNM-rationality remains satisfied (transitivity, in particular) may be difficult.

1.3 Electronic Warfare

Electronic warfare is the strategic use of the electromagnetic (EM) spectrum for militaristic purposes. The field of EW consists of three disciplines: electronic attack (EA), electronic protection (EP), and electronic support [4]. The purpose of EA is to diminish the effectiveness of an opposing radar. EP is utilized to counteract an adversary's EA efforts. Electronic support is a broad term covering applications in EW that do not directly fall under either EA or EP, but is typically for the purposes of gathering information. The EW application discussed in this paper will focus on the jammer's decision-making process. Therefore, this section will focus on describing the EA actions available to the jammer.

For the purposes of this thesis, the goal of a radar is to gather information about a target by actively scanning the environment for EM waveforms. Once a radar discovers a target in the environment, tracking of the target's position and velocity may become possible. Before the radar discovers the target, the objective of the jammer is to deny the radar the ability to detect the target. Once the target is in track, the jammer attempts to

confuse the radar and force it to lose track of the target. For this application, the target is equipped with a self-screening jammer. In other words, the jammer is attempting to prevent itself from being detected and tracked. To protect itself, the jammer has access to three masking techniques – barrage noise (BN), responsive spot noise (RSN), and Doppler noise (DN) – and two deception techniques: velocity gate pull-off (VGPO) and range false targets (RFT).

Masking techniques are designed to subvert the radar's efforts to detect a target [4]. All three of the jammer's masking techniques are noise-based attacks that attempt to prevent a radar from gaining information by overwhelming it with noisy data. BN uses a wide-bandwidth signal to deny as much of the EM spectrum as possible. RSN is similar to BN but is used when the center frequency of the radar's signal is known [5]. Jamming a narrow band centered on the radar's frequency allows the jammer to focus its power, allowing for a stronger, more focused jamming effort. DN is a coherent masking technique that applies phase modulation to the received radar signal [4]. The phase-modulated signal is rebroadcast with the intended effect of denying the radar the ability to determine the target's speed.

Deception techniques attempt to fool the victim radar into believing false information, making it difficult to track the true target [4]. False targets techniques attempt to deceive the radar by modulating and rebroadcasting the radar's signal in order to generate false targets for the radar to track. RFT creates false target returns at many different ranges from the radar, disguising the radar's true position. If the radar cannot distinguish between the real and false targets, it is forced either to track all of them, which is costly, or to track only a few, potentially ignoring the real target. Pull-off techniques

attempt to break a radar's track on a target. These techniques occur in multiple stages. First, the jammer amplifies and rebroadcasts the radar's signal. This action provides the radar with a stronger return than the signal reflecting off the true target and may cause the radar to begin tracking the amplified signal being emitted by the jammer instead. The jammer then begins to modulate the signal so that the return no longer represents the true location or velocity of the target. After some time has passed, the jammer will stop broadcasting the signal. If successful, the radar will have lost track of the target.

1.4 Assumptions and Terminology

In this research, both EA and EP applications are discussed. For both applications, simplifying assumptions have been made. With additional information and analysis, these assumptions could be replaced in order to represent more realistic scenarios.

For the purposes of generality, it is assumed that an agent can operate in one of several modes. For the EW application, the jammer has five modes of operation, corresponding to the five actions it can chose from: BN, RSN, DN, RFT, and VGPO. The emitter also has five modes, where each mode is designed to be particularly effective against a different one of the jammer's modes. These modes are referred to as Anti-BN, Anti-RSN, etc. Agents are also assumed to have access to a library of information that can be referenced in order to make inferences about their opponent's actions. This library should contain models of the different agents (both friendly and unfriendly) that are common in the environment. This information can then be compared to the waveforms received by the agent.

A received signal is compared against each model in the library, and a probability value is assigned to represent how strongly the agent believes its opponent is of the corresponding model. A similar process also computes the probability that the agent's opponent is operating in one of several operating modes. The term state is used to denote the combination of a particular model operating in a particular mode.

For illustration of the non-cooperative game, only a single jammer and a single emitter are assumed to be operating in the environment. As previously mentioned, both the jammer and the emitter are restricted to operate in one of five modes. These modes are the actions that the agent can take. The goal of the jammer is to maximize its utility. The emitter seeks to minimize the jammer's utility by correctly switching into the mode that best defends against the jammer's chosen attack. For the example application of the framework, jammers are only interested in attack and emitters are only interested in self-protection.

1.5 Related Work

There are two main challenges that must be addressed by a wireless decision support framework: the first consists of extracting meaningful information from a received waveform, and the second is processing that information for the purpose of rational decision making. Both of these challenges have been the subject of extensive research in a variety of fields.

1.5.1 Wireless Identification and Classification

The ability to identify and classify objects through exploitation of the electromagnetic spectrum is one of the key problems in the field of wireless communications. A sample of proposed solutions to related applications is presented here, with a focus on radar, but the literature should be consulted for a more comprehensive

understanding of this topic. An overview of basic radar classification concepts can be found in Smith et al. [6].

Gader et al. propose a method of improving the detection of landmines using ground penetrating radar in [7]. Their contribution is the application of hidden Markov models (HMM) to increase the probability of mine detection. In a HMM, data is periodically measured in the form of observations. Intuitively, these observations are measurable features of the modeled system, such as temperature, force, or wavelength, that give probabilistic insight into the unobservable (hidden) state of the system. For example, imagine three identical, closed boxes, one of which contains a ball. Which box contains the ball (the state) is unknown, and the boxes cannot be opened (the state is hidden). However, the boxes can be weighed. Those measurements (the observations) can then be used to infer which box contains the ball. This example is degenerate because one measurement is sufficient to determine the state with high certainty. Usually, HMMs are applied to time-series applications, such as recognition tasks [8][9], where the state of the system changes over time and observations are non-deterministic indicators of the hidden state.

The approach detailed by Gader et al. uses two different HMMs: a background model and a mine model. The background model accounts for the cases when a mine is not present, reducing the chances of false positives. For the mine model, they observed that when a mine is detected the GPR data resembles an inverted parabola, so they chose to use five states: a starting background state, the rising edge of the parabola, the crest, the falling edge, and the ending background state. Over time, the system accumulates observations and, after each observation, renders a Bayesian judgement of the state probabilities for each

model. A likelihood ratio test is then performed between the probabilities for the terminating states of each model, subject to a variable false alarm rate. Their experiments in the field demonstrated exceptional performance in terms of their desired benchmarks.

Efforts in using radar to perform indoor, through-the-wall target classification are typically complicated by the presence of clutter. In [10], a method was proposed to differentiate between indoor clutter and human bodies through the application of a support vector machine (SVM). SVMs are a machine learning technique that use a decision boundary to convert raw data into a binary classification. The decision boundary consists of a separating hyperplane defined by a set of points, known as the support vectors. The algorithm seeks to maximize the margin, or the distance between the support vectors and the decision boundary. This separation is accomplished by iteratively updating the decision boundary until the classifier is optimized for a set of pre-classified training data. Training data consists of features, their values, and a label. Once an SVM is trained, it can be applied to unclassified data in order to generate a classification.

In [10], the radar cross section (RCS), a measurement of how well an object reflects radar waves, was measured for multiple variations of the frequency, aspect angle, and type of polarization, where each combination of these values produced a new feature. Reducing the number of features used to train the SVM is often desirable, which can be accomplished by determining which features best discriminate the data and discarding the rest. Two feature selection algorithms were used for this purpose. The first ranks features by their Fisher score, which is a measure of how much information that feature provides. Thresholding is then used to select the best features. The second uses the RELIEFF algorithm to compute a weight vector whose entries correspond to each of the features. The

weights are updated iteratively according to how close a given feature vector is to the closest members of each class. Features that have many, identically-classified instances near each other will have higher scores. Likewise, features with many instances classified far away from oppositely-classified instances will have higher scores. Although the paper does not state so explicitly, the best features are presumably selected using thresholding, as before.

The approach proposed in [10] was tested for both simulated and experimental data, for each of the feature selection algorithms in addition to the full feature set, for linear and RBF kernels, and for scenarios with and without noise introduced from various types of walls, including drywall, plywood, and brick. The success of the classifier was heavily dependent on the SNR: weak signals failed to outscore random classification, while strong signals often resulted in classification accuracy of over 90%. Comparisons of performance between kernels and styles of polarization are difficult because the axes are inconsistently spaced, but the RBF kernel appears to outperform the linear kernel in most – if not all – trials, while both styles of polarization scored similarly on the simulated data. For experimental data, the SVM trained on data captured using vertical, co-polarization outperformed the SVM trained on data captured using horizontal, co-polarization.

1.5.2 Game Theory

Since being introduced, game theory has seen extensive use as a tool for analysis, autonomous decision making, and optimization in virtually every academic field, including economics [3], engineering [11], logistics [12], robotics [13][14], ecology [15][16], and many others. The field of wireless communication is no exception, and game theory has a long history of use in this area.

For example, the authors of [17] used game theoretic principles to design a distributed, adaptive system that controls the transmission power of a wireless device in a manner that maximizes energy efficiency while ensuring a consistent quality of service. Other devices in the network are viewed as competitors and are tracked using a HMM, where the hidden states are the competitors' wireless channels and where an agent's decision corresponds to the amount of power used to transmit the signal. This scheme allows a wireless device to determine if a channel is empty, even in the absence of complete information. The utility of each agent's action is expressed in terms of the agent's desire to increase its signal to interference plus noise ratio (SINR) (by increasing the transmit power) weighed against the cost of the transmission. By combining these two factors in the form of a non-cooperative game, each agent is able to adaptively choose the channel that minimizes their individual energy consumption while achieving at least the minimal quality of service. The authors were able to show the success of their approach through a simulated game, which resulted in increased efficiency, in terms of the capacity to power ratio, at the cost of implementation complexity.

In [18], Doshi et al. show how game theory can be applied to resolve denial of service (DOS) attacks against wireless ad hoc networks. Specifically, the problem of a gray hole attack is discussed in which an intelligent adversary selectively drops packets in order to avoid detection. The solution to DOS attacks typically consists of removing the malicious node(s) from the network, but, during a gray hole attack, distinguishing a node that is unreliable from one that is malicious is difficult. Their paper provides a policy of optimal protection against such attacks through a game theoretic description of the problem, where the sender competes against a malicious node in a simple, non-cooperative

game. The sender of a packet tracks the reliability of nodes in the network as true detection and false alarm rates that are updated over time according to whether the packet was ultimately delivered or dropped. These probabilities are then used to determine the utility of using the corresponding node and the utility for dropping the node from the network. Using a Nash Equilibrium analysis, the authors determine the conditions under which the sender should drop a node and under which a malicious node should risk dropping a packet. However, it should be noted that the success of their approach was not completely demonstrated, nor was it compared against alternative methods of resolving DOS attacks.

While non-cooperative game theory is often a useful tool for determining resource allocation in wireless applications, cooperative solutions have also been successful in the past. For example, in [19], Béjar et al. discuss a scheme that uses coalitional game theory to reduce resource consumption during target localization. Localization algorithms commonly combine distance measurements from multiple wireless sensor networks (WSN) in order to achieve more reliable tracking information than a node could obtain alone. Typically, measurements from every communicable WSN are used in an effort to maximize the reliability of the resulting estimate. However, the authors suggest that limiting the number of participating WSNs could extend resources without sacrificing accuracy, and they offer a solution that groups WSNs into coalitions, which are determined by weighing the costs of communication against the benefits of extra information. The members of these coalitions can then share information amongst themselves, benefiting the collective group. The utility of a coalition is determined by assessing the coalition's quality against the communication cost of formation. Quality is a function of the summed, squared error between the individual position estimates and the group's consensus. Coalitions are

formed through successive steps of merging and splitting existing coalitions, which continue until the coalitions converge. The process of selecting coalitions repeats after a fixed interval, starting from the set of coalitions previously determined, and, after each convergence, the coalition with the highest utility performs the localization task while the WSNs from the other coalitions become inactive to conserve energy. While Béjar et al. do manage to significantly reduce energy expenditure with their approach, the localization error approximately doubled (from 0.34 m), and how these results might change for different sleep intervals or target motion models is unclear from their work.

CHAPTER 2

A General Decision Framework

This chapter describes the function and purpose of each of the general framework's components. First, a high-level overview is provided. Then, a structure known as the classification matrix is introduced. This chapter concludes with an analysis of each of the three main modules, given in the order they occur during the decision making process.

2.1 Overview of the Decision Framework

As shown in Figure 1, the framework consists of three modules: the adversary identification module, the utility calculator, and the optimizer. The modules act sequentially and each is dedicated to a different step in the decision making process. The adversary identification module is responsible for processing environmental data. While it

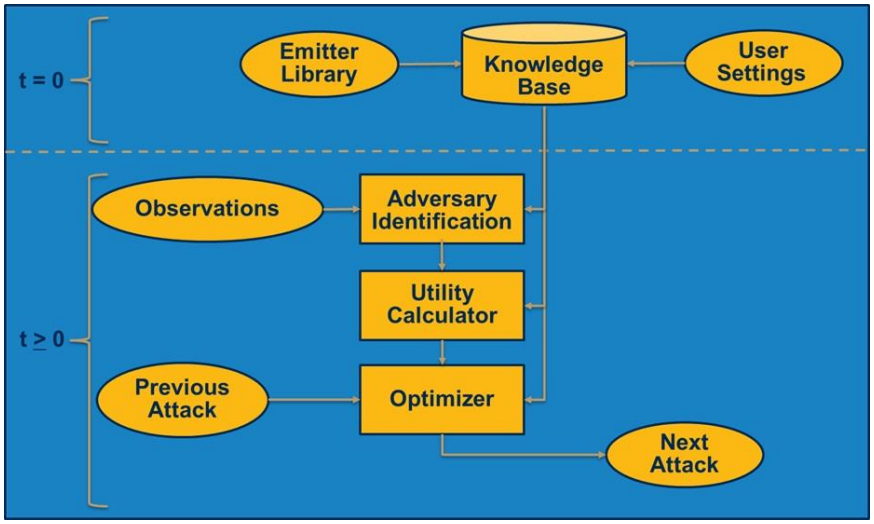

Figure 1. This figure shows a complete overview of the decision framework. The adversary identification module uses observed data and information from the agent library stored in the knowledge base to compute the classification matrix for use in the utility calculator module. The expected utilities for each action are then passed to the optimizer to rank the agent's potential actions.

is not strictly necessary for the framework to function, the information it provides to the utility functions increases the accuracy of the utility evaluation. The utility calculator module is responsible for evaluating the potential effects of an agent's actions and is the most critical part of the framework. Finally, the optimizer sorts the agent's responses based on those utility and cost values as well as user-defined constraints.

The first module, adversary identification, begins by analyzing the received waveforms in order to determine if they were emitted by an agent. If so, the adversary identification module attempts to determine the state of the detected agent. It does this by determining the likelihood of receiving the observed data given that the opponent is in a given state, s, for each possible state combination. The probabilities for each state are stored in a structure referred to as the classification matrix. Using the classification matrix and user-defined utility and cost functions, the utility calculator estimates the expected utility and cost of each response through a multi-step process. The utility calculator first evaluates the utility and cost of each response for each state combination. Then, it weights the utility and cost for each state by the probability of that state's occurrence. The sum of the weighted utilities yields the average utility a response can be expected to produce, known as the expected utility of the response. The optimizer is responsible for selecting the action an agent will take. This is accomplished by ranking the actions available to the agent from most to least preferable. Preference is determined from a combination of the expected utility and cost of each action as specified by user-defined optimization criteria.

2.2 Classification Matrix

The classification matrix stores the joint probability of each state as computed by the adversary identification module, and its configuration is shown in Figure 2. The classification matrix provides a means of tempering the agent so it does not overvalue unlikely states or discount low-value states.

For each other agent in the environment (friend or foe), the adversary identification module determines the likelihood that the received waveform was generated by each of the models in the agent's library and also estimates the likelihood that the waveform was generated in a certain mode. The rows of the classification matrix account for the modes that the detected agent could be operating in while the columns account for the various models in the library. The final column is, optionally, used to account for the possibility that the correct model is not in the library. The intersection of a column and a row is a state, and the value of the classification matrix for that state is the likelihood of that state occurring given the observed data. Since the detected agent must be operating in some

		Agent Models in Library			
		AM_1	…	AM_n	Unknown Model
Mode of the Opponent	M_1	$P(M_1\|AM_1)P(AM_1)$	…	$P(M_1\|AM_n)P(AM_n)$	$P(M_1\|U)P(U)$
	⋮	⋮	…	⋮	⋮
	M_m	$P(M_m\|AM_1)P(AM_1)$	…	$P(M_m\|AM_n)P(AM_n)$	$P(M_m\|U)P(U)$

Sums to 1

Figure 2. This figure is the general form of the classification matrix. Each index of the matrix represents a different state. The rows represent the different modes of operation in which the opponent could be operating, and the columns represent the different models for the opposing agent. The value of the classification matrix at a given index is a probabilistic estimate that the opponent is in that state.

mode and must belong to a model type (even if that model is unknown), the classification matrix for each detected agent in the environment is constrained to sum to 1. If it is assumed that the library of models is comprehensive (i.e. that there are no unknown models), then the classification matrix is normalized to satisfy this constraint, thereby providing the joint probability of each state. These probabilities are later used as weights for the outputs of the utility calculator for the purpose of calculating the expected utility for a given action.

2.3 Adversary Identification Module

The adversary identification module's main responsibility is to compute the classification matrix. This process is heavily reliant on a library of models containing information about the different types of agents that may be present in the environment. This information may include averages or ranges for a large number of parameters. If models are not available, the performance of the framework will suffer because the agent will not be able to make informed decisions. This heavy reliance on models could be viewed as a potential weakness of the framework.

In order to compute the classification matrix, the adversary identification module compares the received waveforms to the models in the library and attempts to determine which model best matches the observed waveform. This task can be viewed as a classification problem that can be solved using various unsupervised machine learning techniques, such as Gaussian mixture models (GMM) [20] or support vector machines (SVM) [21]. If training data is readily available, supervised machine learning algorithms can also be applied. Because agent models were not available for the electronic warfare

application presented in Chapter 3, the classification matrix was implemented using an auto-regressive model. The details of the implementation are given in Section 3.3.

2.4 Utility Calculator

Using the classification matrix and a set of utility functions, the utility calculator determines the expected utility of each action. Recall that the utility functions are the method through which an agent quantifies the benefit received from taking an action. Utility functions are typically, but not necessarily, dependent on both the state of the opponent, s, as well as environmental factors, θ. The expected utility, U, of an action, a, is found in three steps. First, the action's utility function is evaluated for each state, s. Then, the utility values of each state are weighted by the corresponding probability stored in the classification matrix. Finally, the weighted utilities are summed over all possible states. The expected utility can be compactly expressed as

$$U_a(\theta) = \sum_{s \in S} u_a(s, \theta) p(s), \tag{1}$$

where S is the set of all opponent states, u_a is the utility function for action a, and $p(s)$ is the probability of state s. The probabilistic weighting prevents the framework from over-valuing unlikely yet high-utility states and results in rational behavior as long as the set of utility functions satisfy the VNM rationality requirements. If applicable, the utility calculator can also compute the expected cost of an action using an analogous set of cost functions in place of the utility functions in (1). The resulting expected cost can, optionally, be used as another constraint during the optimization step, as detailed in the following section.

2.5 Optimizer

The final module ranks the options available to the agent. There are many different, yet valid, ways to rank an agent's responses. Without constraints, the most logical approach would be to rank the actions from highest expected utility to lowest. Otherwise, cost constraints can be used to prohibit the agent from taking actions that are too expensive. Additionally, when multiple responses can be performed simultaneously, responses can be ranked according to the highest combined utility subject to some cost threshold. A more complex optimization strategy could combine dissimilar actions in order to maximize worst-case performance. Ultimately, the optimization approach is determined by user preference.

When the optimizer is combined with the adversary identification and utility calculator modules, the resulting system is capable of choosing rational actions based on observed data and user-defined constraints. The following chapter will provide an example of how the general framework can be applied to the problem of providing decision support for a jammer and an emitter in an EW environment.

CHAPTER 3

An Electronic Warfare Application

This chapter details how the general framework was implemented to provide decision support for the example problem of an EW conflict between a jammer and an emitter. Section 3.1 reintroduces the problem and underlying assumptions and gives an overview of how the classification matrix was implemented. The implementation of the adversary identification module is discussed in Section 3.2. Finally, the design and rationale behind the jammer's utility functions are provided in Section 3.3.

3.1 Framework Implementation

The efficacy of the framework is shown through an example problem consisting of a non-cooperative game in an EW environment. As mentioned in Chapter 1, several assumptions are made to limit the complexity of the example problem. First, the number of agents in the environment is limited to a single emitter and a single radar jammer. Second, the agents behave antagonistically toward each other and both seek to optimize their own expected utility. Third, the agents are limited in the number of actions they can take. Fourth, during the game, the actions of each agent are determined using its respective implementation of the framework discussed in Chapter 2.

There are several different approaches that could be used to calculate the classification matrix. A standard approach is to use a recursive Bayesian estimator, such as a particle filter, which updates the distribution (i.e. the classification matrix) after each measurement such as in [22]. This approach marginalizes the degree to which noisy

measurements skew the distribution. Another option views the problem as a type of multi-class classification, wherein data from the environment are compared against agent models to find the best match. In [23], neural nets were used to classify a radar using two separate case studies. The first case study was able to determine if a radar was hostile, while the second was able to accurately classify the radar's mode. In [24], Liu et al used clustering methods to classify radar returns. Using their algorithm, they were able to correctly determine which radar emitted a signal with 93% accuracy. Section 3.2 details how the classification matrix is generated and updated for the EW application.

In Figure 3, the general form of the classification matrix is shown for a jammer with *n* emitter models and *m* detected emitters. Each entry is the joint probability that the waveform emanated from an emitter of a specific model type while operating in a certain mode. The emitter models are denoted as EM_j and shown in the columns. The models represent different types of emitters that could be found in the environment. Each block of rows, E_i, represents the i^{th} emitter agent detected in the environment. Each row inside a block represents one of the different operating modes: Anti-BN, Anti-RSN, Anti-DN, Anti-

			Emitter Models in Library				
			EM_1	…	EM_n	Unknown Model	
Mode of the Opponent	E_1	M_1=Anti-BN	$P(M_1\|EM_1)P(EM_1)$	…	$P(M_1\|EM_n)P(EM_n)$	$P(M_1\|U)P(U)$	Sums to 1
		⋮	⋮	…	⋮	⋮	
		M_5=Anti-RFT	$P(M_5\|EM_1)P(EM_1)$	…	$P(M_5\|EM_n)P(EM_n)$	$P(M_5\|U)P(U)$	
	⋮	⋮	⋮	…	⋮	⋮	
	E_m	M_1=Anti-BN	$P(M_1\|EM_1)P(EM_1)$	…	$P(M_1\|EM_n)P(EM_n)$	$P(M_1\|U)P(U)$	Sums to 1
		⋮	⋮	…	⋮	⋮	
		M_5=Anti-RFT	$P(M_5\|EM_1)P(EM_1)$	…	$P(M_5\|EM_n)P(EM_n)$	$P(M_5\|U)P(U)$	

Figure 3. The general layout of the Classification Matrix for the jammer. Each column, EM_j, corresponds to a different emitter model. Each block of rows, E_i, represents the i^{th} emitter detected in the environment. Each row inside a block represents a different operating mode: Anti-BN, Anti-RSN, etc. An element in the matrix is the probability that the corresponding emitter is operating the given state.

RFT, and Anti-VGPO. In general, however, the matrix does not have to be limited to these five modes; rather, it can be expanded to include different modes for each emitter. An additional column is added to account for the cases where the adversarial model may not be in the library. As shown in Figure 3. The elements of the matrix are constrained to sum to *m*. This constraint comes about because there are *m* agents, and the conditional probabilities for each agent summed across all models and modes is one. Using this condition, the probability of an unknown adversarial model can be determined by subtracting the sum of the other columns from 1. The information in Figure 3 is valid for either an emitter or a jammer classification matrix; they differ only in which modes an agent can utilize and the models in the library.

The design of the utility functions are the most critical aspect of the framework implementation because they control how the agent evaluates its actions. Even if the rest of the framework is operating perfectly, inaccurate or poorly designed utility functions may result in irrational decisions. For the jammer, five utility functions were developed to calculate the utility for five techniques: BN, RSN, DN, RFT, and VGPO. For the emitter, analog utility functions were constructed to counter each of the corresponding jamming attacks. The utility functions were constrained to be functions of only a single variable in order to reduce the complexity of the analysis.

BN and RSN are similar techniques and, thus, base their utility function calculation on an estimated Jamming-to-Signal Ratio (JSR) value. Since RSN will use a lower bandwidth than BN, the JSR for RSN should have a higher utility value than BN, if both attacks use the same amount of energy. For DN, the utility is computed based on the difference between the Signal to Noise Ratio (SNR) and the SINR. For RFT, the jammer

estimates the number of false targets it can generate within a single dwell. The utility is then based on the ratio between the number of false targets and the maximum number of targets the emitter can simultaneously track. The utility for the VGPO attack is based on the amount of time that the jammer estimates it can deceive the emitter into following a false track. Each emitter is assumed to be able to recognize a false track once the target and the false track are sufficiently separated. Therefore, the amount of time the jammer can mislead the emitter is calculated as a function of the velocities of the target, the false track, and the minimum distance at which the emitter can identify the subterfuge. Both RFT and VGPO require information that is dependent on the model of the emitter. This information is assumed to be provided in the library of models.

Table 1. This table gives the values of the emitter's utility functions. The utility functions are dependent on the jammer's mode. The rows each represent one of the emitter's utility functions. The columns each represent a different jammer action (i.e. one of the jammer's modes).

	BN	RSN	DN	VGPO	RFT
Anti-BN	0.675	0.525	0.375	0.000	0.000
Anti-RSN	0.525	0.675	0.375	0.000	0.000
Anti-DN	0.375	0.375	0.600	0.000	0.000
Anti-VGPO	0.000	0.000	0.000	0.975	0.675
Anti-RFT	0.000	0.000	0.000	0.675	0.975

Since the behavior of the emitter is not the focus of the research, the emitter utility functions were not a primary focus in the implementation of the game. The emitter utility functions depend only on the mode of the opponent and are listed in Table 1. Each mode is designed to counteract one of the jammer's attacks. It is also assumed that the modes also provide utility against attacks similar to the one it counters. For example, BN, RSN, and DN are all masking jamming techniques, so Anti-BN is effective against each of them. Likewise, VGPO and RFT are both deception techniques, so Anti-VGPO is effective against both them. The utility of the deception countering modes is higher, on average, than

the utility of the modes that counter masking techniques because the deception attacks are stronger threats.

Originally, instead of the utility function returning a constant value for a given jammer mode, the emitter's utility was a random variable constrained on a small interval. However, this randomness violated VNM rationality (i.e., completeness did not hold) and caused the emitter to choose actions irrationally. With mode-dependent utility functions, the emitter tends to select the action that counteracts the most likely jammer mode, but, because the emitter is maximizing its expected utility, it may choose to counteract a mode other than the one which is most likely. Since the emitter has limited knowledge of the opponent's state, it will not always be able to select the optimal response. This problem – reasoning in the presence of uncertainty – is the motivation behind using a game theoretic approach for decision support. Optimizing over expected utility allows the agent to maximize its average case performance.

The total cost for a jamming attack is determined by computing four values which are then weighted and summed together: the percent of the maximum available power used for the attack, the amount of energy expended by the attack, the opportunity cost of using the attack, and the "detectability" of the attack. In general, the weights assigned to the individual costs are based on user preference for a specific application. Since the optimizer was implemented to choose the attack with the highest utility, the cost functions do not contribute to the decisions made by the agent in the simulation. As a result, the optimizer will always sort the actions from best to worst in the order of decreasing utility, regardless of cost.

3.2 Implementation of the Adversary Identification Module

The classification matrix is updated over time to account for new data using an auto-regressive model. Because of the difficulty in obtaining authentic EW data and agent models, data were randomly generated based an agent's action. At each time step, an agent receives an observation that is used to determine a measurement of the opponent's previous action. The agent receives the correct measurement with probability p. If the agent receives a wrong measurement, it is equally likely to receive a measurement of each of the other modes. In general, the value of p may differ between agent models. As such, the measurements themselves may be different for each model. If mode i was measured, the corresponding measurement vector, m, would be expressed as

$$m(j) = \begin{cases} 0, & j \neq i \\ 1, & j = i \end{cases}. \tag{2}$$

The matrix M_t is composed of the column-vector measurements at time t for each of the agent models. The classification matrix is updated over time according to

$$C_t = (1 - \gamma)C_{t-1} + \gamma M_t, \tag{3}$$

where C_t is the classification matrix at time t and $0 \leq \gamma \leq 1$ is the learning rate. The learning rate adjusts how quickly the agent responds to new data. A high learning rate is sensitive to noise, while a low learning rate will cause the agent to take longer to respond to the opponent. For the agent models in the simulation, a value of 0.8 was used for p. The jammer's learning rate was set at 0.1; the emitter used a value of 0.25 for the learning rate. The emitter requires a higher learning rate because its utility functions are more mode-dependent than the jammer's, and, as a result, the classification matrix has more influence over the final decision.

3.3 Implementation of the Utility Functions

For the utility values to be consistent across attacks, the parameters (JSR, the number of false targets, etc.) discussed in Section 3.1 are weighted, shifted, and passed through a logistic function, given by (4). This process accounts for the dissimilarities between the parameters and maps the utility onto the unit interval, thus allowing for the utility values to be easily compared and analyzed. The utility functions are of the following form:

$$u(\theta) = \frac{1}{1 + e^{-\alpha(\theta - \mu)}}, \tag{4}$$

where α and μ are tuning parameters used to account for inconsistencies in the distributions of the parameters. The problem with this approach is that the values of α and μ are subjective and potentially difficult to balance with respect to the other utility functions. This problem of determining α and μ conflicts with the modular design of the framework, because it becomes more difficult to add or remove utility functions. To address this, a process was developed to solve for α and μ in a less subjective way. For each attack, two inputs (typically upper and lower bounds) were chosen and assigned values. Let $\theta_1$and θ_2 be the inputs to a utility function, and let $u_1$and u_2 be their respective desired outputs. Then, a logistic function of the form (4) can be found to uniquely satisfy these points where the values of α and μ can be expressed as

$$\alpha = \frac{c_1}{(\theta_1 - \theta_2)} \tag{5}$$

$$\mu = \frac{c_2\theta_1 - \theta_2}{c_2 - 1}, \tag{6}$$

where

$$c_1 = \ln\left(\frac{{u_2}^{-1} - 1}{{u_1}^{-1} - 1}\right) \tag{7}$$

$$c_2 = \frac{\ln({u_2}^{-1} - 1)}{\ln({u_1}^{-1} - 1)}. \tag{8}$$

Each attack uses a different parameter for θ in (4). Barrage Noise and Responsive Spot Noise, use JSR for θ, which is given by:

$$\text{JSR} = \frac{P_j G_j d_j 4\pi R^2}{P_t G_t B_j \sigma n \tau}, \tag{9}$$

where P_j and P_t are the peak transmit power of the jammer and the emitter, respectively; G_j and G_t are the transmit gains of the jammer and the emitter, respectively; d_j is the duty cycle of the jammer; R is the distance between the jammer and the emitter; B_j is the bandwidth of the jammer; σ is the RCS of the target which, in this case, is the jammer, because the jammer is self-screening; n is the number of pulses emitted by the emitter; and τ is the pulse width of the emitter's signal [4]. The JSR was chosen because it directly relates to the probability of detection of a target. Since the objective of the jammer is to minimize the probability of detection, ideally the probability of detection would be used in the utility function. However, the probability of detection itself may not be observable in some scenarios. The JSR is a natural choice as a substitute for probability of detection because the probability of detection decreases as the JSR increases.

For DN, the argument of the utility function was chosen to be the difference between the SNR and the SINR, which are given by the equations

$$\text{SNR} = \frac{P_t G_t G_r \lambda^2 \sigma n \tau}{L_s (4\pi)^3 R^4 k T F} \tag{10}$$

and

$$\text{SINR} = \frac{P_t G_t G_r \lambda^2 \sigma n \tau}{L_s (4\pi)^3 R^4 \left(\frac{P_j G_j \lambda^2 B d_j}{4\pi^2 R^2 B_j} + kTF \right)}, \tag{11}$$

where λ is the signal wavelength, L_s is the emitter's system losses, B is the emitter's bandwidth, k is Boltzmann's constant, T is the reference noise temperature, and F is the emitter's receiver noise factor [4]. The difference on a linear scale between (10) and (11) provides an estimate of the effectiveness of the jammer's attacks. The SNR value is an indicator of how well an emitter can detect objects in the environment, and the SINR value is a similar estimate but includes the presence of interference. The difference between the two values can be interpreted as a measure of how effectively the jammer inhibits the emitter from detecting targets in the scene.

For both RFT and VGPO, an assumption is made that the emitter has limitations that are known ahead of time and stored in the knowledge base's library. These limitations are used to determine the values of the parameters input into the utility functions. The RFT utility function assumes that the emitter can only keep track of at most N_{max} targets. The value of N_{max} may be different for each emitter model but is assumed to be known a priori. The VGPO utility function assumes that the emitter cannot distinguish between the true target and a false track until they are a distance of d away from each other. Again, the value of d can differ between emitter models but must be known ahead of time.

When a jammer utilizes RFT to attack an emitter, the emitter is forced to respond in one of two ways. The emitter can either simultaneously track all of the received returns, or, when the number of potential tracks exceeds the maximum number of tracks that the emitter can maintain, it must decide which returns to track. Therefore, the RFT utility function has two methods of gaining utility. The first is by consuming the emitter's

resources by forcing it to track false targets. The second is by decreasing the probability that the correct target is selected to be tracked.

For each false track that the emitter decides to track, the jammer is awarded with a small amount of utility, ϵ. Since the jammer does not know which false tracks the emitter is tracking, the jammer assumes each track is accepted with probability P_a. Therefore, if the jammer generates N_{FT} false tracks, it is rewarded with a utility totaling $u_{RFT} = \epsilon P_a N_{FT}$. The amount of utility gained by forcing the emitter to track a false target cannot exceed $\omega = \epsilon N_{max}$ because the emitter cannot track any additional returns and, after that point, generating additional tracks would not cause the emitter to expend additional resources. The value of ω acts as a ceiling on the amount of utility that the jammer can accrue by forcing the emitter to expend resources. The value of ω is required to be less than one (a value of 0.3 was used in this example for ω). Note that assigning a value to ω also determines the value of ϵ. However, since each emitter model may have a different value for N_{max}, the value of ϵ may be different for each emitter. The jammer estimates how many false tracks it can produce using

$$N_{FT} = f\left(\frac{1}{PRF} - \frac{R}{c}\right), \tag{12}$$

where *f* is the maximum number of false targets the jammer can generate per second, *PRF*, is the pulse repetition frequency of the emitter, *R*, is the range between the jammer and the emitter, and $\boldsymbol{c}$ is the speed of light.

Even though producing more than N_{max} false targets does not cause the emitter to expend additional resources, producing additional false targets provides a secondary benefit. For the purposes of this simulation, the emitter is constrained to track at most N_{max} returns, and it is assumed that each track has an equal probability of being selected.

When the emitter cannot track all of the returns it receives, a choice must be made as to which returns will be tracked. When the emitter is forced to choose which returns will be tracked, it may stop tracking critical targets. As the number of false targets increases, the probability that the emitter stops tracking a real target increases. Let N_T represent the number of targets the jammer is protecting, and let N_O represent the number of other returns. The utility function rewards the jammer based on the following parameter:

$$\theta_{RFT} = \frac{P_a N_{FT} + N_T + N_O}{N_{max}}. \tag{13}$$

Since N_O is unknown, it is assumed to be the worst case of 0 (larger values of θ_{RFT} result in higher utility).

The parameter defined by (13) is the argument for equation (4) for the RFT utility function. When $\theta_{RFT} < 1$, which implies that the emitter is not saturated ($P_a N_{FT} + N_T < N_{max}$), the utility function is linear with a slope of ϵ. When $\theta_{RFT} = 1$, the jammer is consuming the maximum amount of the emitter's resources, so it has utility of ω. When $\theta_{RFT} > 1$, the utility function grows exponentially starting from ω with an upper bound of one. The utility function is expressed as

$$u(\theta_{RFT}) = \begin{cases} \omega\theta_{RFT}, & \theta_{RFT} < 1 \\ \omega^{\theta_{RFT}^{-1}}, & \theta_{RFT} \geq 1 \end{cases}. \tag{14}$$

A graph showing a plot of the utility function for $\omega = 0.3$ is shown in Figure 4.

Since the aim of RFT is to consume an emitter's resources over a long period of time, a penalty term was applied to the utility function to punish short duration jammer attacks. The penalty term has the form

$$w(t) = 1 - e^{-t/\mathrm{T}}, \tag{15}$$

where t is a user defined parameter that specifies how long the RFT attack should be run before searching for a new optimal attack and T is a tuning parameter which sets the rate at which the penalty decays. The final utility is then given by

$$u(t, \theta_{RFT}) = w(t)u(\theta_{RFT}). \tag{16}$$

The utility function for the VGPO attack estimates the amount of time it will take before the true target and the false target generated by VGPO separate enough for the emitter to be able to differentiate between them. Given that the minimum distance at which the emitter can make this distinction is known (d), the only information the jammer requires is the speed at which the protected target is moving, the speed of the false track, and the orientation between the two. Note that a self-screening jammer protects itself.

Let the velocity of the target in three dimensions be represented by the vector v_T and the velocity of the false track likewise be represented by v_{FT}. Both of these vectors lie in a standard Euclidean frame centered on the target and are oriented such that the direction

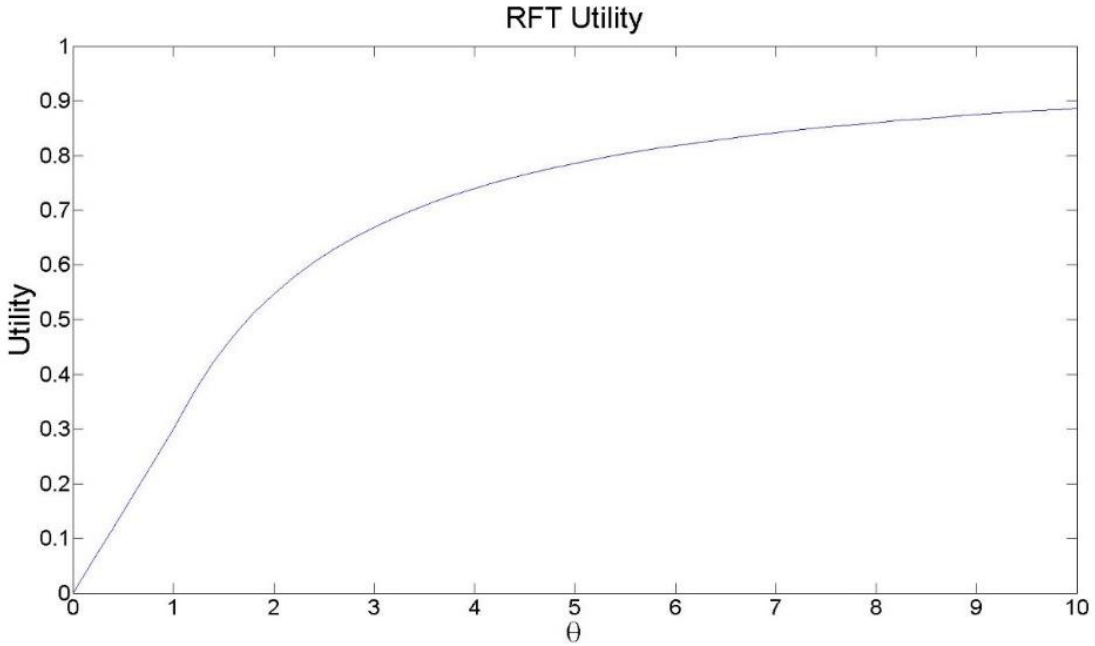

Figure 4. A plot of the utility function for RFT over θ before the penalty term is applied. This plot uses ω= 0.3. On the interval 0<θ<1, the function is linear. For θ≥1, the function is exponential and has a limit of one as θ approaches infinity.

of the target lies along the x-axis. Then, $v_T = \begin{bmatrix} v_{Tx} \\ 0 \\ 0 \end{bmatrix}$ and $v_{FT} = \begin{bmatrix} v_{FTx} \\ v_{FTy} \\ v_{FTz} \end{bmatrix}$. Also, let $v_{FTxy} = \sqrt{v_{FTx}^2 + v_{FTy}^2}$, $v = \begin{bmatrix} ||v_T||_2 \\ ||v_{FT}||_2 \end{bmatrix}$, and the angles between the orientation of v_T and v_{FT} in the xy and yz planes as ϕ_{xy} and ϕ_{yz}, respectively as

$$\phi_{xy} = cos^{-1} \frac{v_{FTx}}{v_{FTxy}} \tag{17}$$

and

$$\phi_{yz} = cos^{-1} \frac{v_{FTxy}}{||v_{FT}||_2}. \tag{18}$$

The amount of time it takes for the target and the false track to separate by a distance d is given by

$$t = \frac{d}{\sqrt{v^T A v}}, \tag{19}$$

where the symmetric matrix A is calculated as

$$A = \begin{bmatrix} 1 & -\cos(\phi_{xy})\cos(\phi_{yz}) \\ -\cos(\phi_{xy})\cos(\phi_{yz}) & 1 \end{bmatrix}. \tag{20}$$

The resulting value of t from (19) was used as the argument for the VGPO utility function. Larger values of *t* represent more effective VGPO attacks and higher utility states. However, this choice for a parameter has a clear flaw. From (19), the optimal outcome (utility = 1) for the VGPO utility function results when the target and false track both move at the same rate in the same direction. This results in an estimate that VGPO can be used for an infinite amount of time without the emitter detecting the subterfuge. However, since the target and the false track are not separating from each other, the false track and the jammer are both in the same location, and the jamming attack is completely ineffectual

because the emitter is not being pulled away. In fact, it would be better for the jammer not to do any attack in this case because VGPO provides an even stronger return and would only serve to act as a beacon broadcasting the jammer's exact location. This flaw is a result of the simplicity of the utility function and could easily be accounted for by using a utility function that considers multiple parameters.

In addition to the dependence on the parameters mentioned above, each jammer utility function is also dependent on the mode of the emitter. Intuitively, this is because the jammer is trying to maximize the success of its actions and does not want to be operating in the mode that is being countered by the emitter. The final utility for the action a against state s is given by

$$u_a(s,\theta_a) = \begin{cases} 0.4u(\theta_a), & \text{if operating mode counters } a \\ u(\theta_a), & \text{otherwise} \end{cases}. \quad (21)$$

Without state-dependent utility functions, the jammer cannot adapt to the emitter's behavior. Furthermore, the classification matrix would serve no purpose because every state would provide the same utility. Also, because the utility functions have an upper bound ($u_a \leq 1$), the jammer will always be able to adapt its actions in response to the emitter as long as there is a viable alternative. For example, the JSR for the RSN and BN utility functions scales exponentially with the range between the jammer and the emitter, but the utility function itself cannot return a value greater than one. In the simulation, because of the upper bound on the utility function, the jammer will switch from RSN to VGPO for any arbitrary value of the JSR if it is sufficiently certain that the emitter is using Anti-RSN.

This chapter has shown how the classification matrix, adversary identification module, and the utility functions were implemented in order to model the decision making

process for an agent engaging in EW. The following chapter will show the effectiveness of the implemented framework through the results of a simulated game in an EW environment. The results from the simulation show that both agents perform rationally when using the implemented framework for their decision-making processes.

CHAPTER 4

Simulation Results

In order to demonstrate the framework, a non-cooperative game was designed and simulated using MATLAB®. The primary purpose of this game was to evaluate the performance of the framework and to analyze the behavior of the jammer under different initial conditions. The game was established as a competition between two agents – a jammer and an emitter – and the only significant difference between the two agents was the set of utility functions each agent utilized. During the game, the objective of each agent was to maximize its own utility. For well-designed utility functions, this behavior would result in an agent choosing the most rational action. For the jammer, this action was typically the attack that was most effective against the emitter's presumed state. The emitter was most effective when it correctly matched defensive strategy to the jammer's attack type.

Four simulations were run under different initial conditions. The first simulation was used as a baseline for comparison with the other three simulations. The second simulation focused on decreasing the utility of the most frequently selected attack (RSN) to observe how the jammer would adapt its strategy. The third and fourth simulations increased the utility for a different jamming attack, VGPO and RFT, respectively. The utility of VGPO was increased by decreasing the velocity of the jammer and its false track. Using (19), this change resulted in an increase in the value for t, thereby increasing the utility. The utility of RFT was increased by raising the number of false targets that the jammer can produce per second (FTPS). Each simulation consisted of twenty games, each

of which lasted for twenty time steps. Across all twenty time steps, the range between the jammer and the emitter was held constant, but after each game finished, the range between the two agents was increased for the subsequent game. Each simulation used the same set of range values, which were chosen to be linearly spaced between 0.1 km and 20 km. At each time step, the jammer and the emitter each chose an action according to the expected utilities calculated during that time step. Each simulation consisted of a total of 400 decisions per agent per simulation.

Table 2. These values are used for the parameters in the simulation and were chosen based on the values provided in [4].

Emitter Parameter	**Value**	**Jammer Parameter**	**Value**
Transmit Power (P_t)	200 kW	Transmit Power (P_j)	200 W
Bandwidth (B)	10 MHz	Bandwidth for BN (B_{BN})	100 MHz
Gain (G_t)	30 dBi	Bandwidth for RSN (B_{RSN})	75 MHz
Wavelength (λ)	0.05 m	Bandwidth for DN (B_{DN})	10 MHz
Pulse width (τ)	2 ns	Gain (G_J)	15 dBi
Number of pulses (n)	16	System noise figure (F)	5 dB

The parameters used to calculate the JSR, SNR, and SINR were taken from [4] and are shown in Table 2. Recall that the JSR, given by (9), was used in the utility calculation for the BN and RSN attacks and that the SNR and SINR, given by (10) and (11)(10) respectively, were used in the utility calculation for the DN attack. Note that, since BN attacks use larger bandwidths than RSN attacks, BN should have a lower JSR than RSN when all of the other parameters are held equal. As a result, the utility for RSN should always be higher than the utility of BN against a single opponent.

The jammer's decisions for each of the four simulations are shown in Figure 5, and the emitter's decisions are shown in Figure 6. Each decision is represented by a colored block in the figure where each color represents a different jamming attack or emitter action. For a given point, (x, y), the figures show the x^{th} decision that the agent made for the game

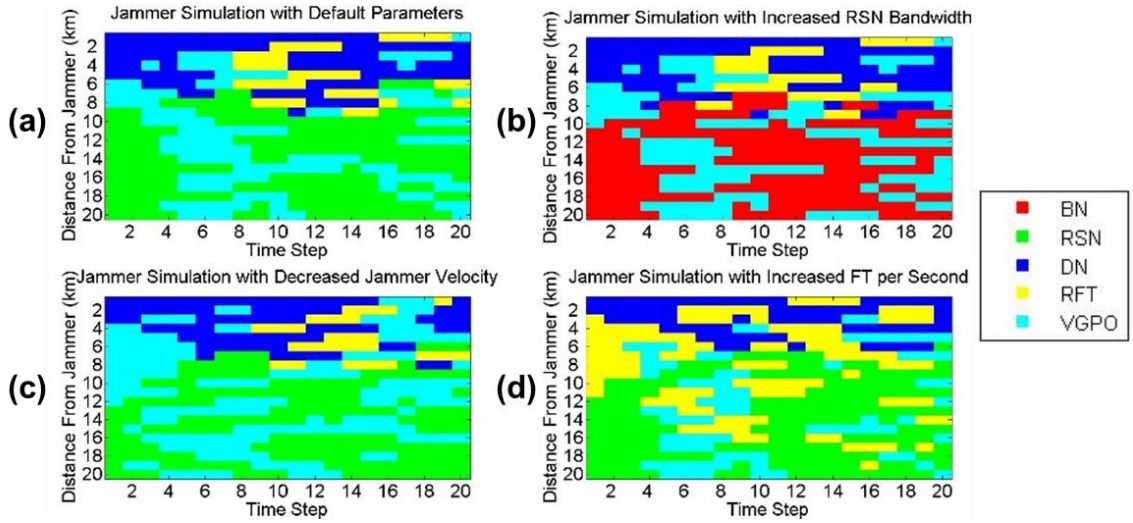

Figure 5. The jammer's decisions are shown for all distances and time steps for each simulation. The first simulation (a) acts as a baseline for comparison against the other three simulations. The remaining three simulations show the effects of increasing the utility for each of the non-dominant decisions: BN, VGPO, and RFT, respectively.

that was simulated for a range of y km between the two agents. Note that the y-axis is inverted with the short-range games at the top and the long-range games at the bottom.

The most interesting aspect of each graph in Figure 5is the way the agents changed their actions over time. For example, as shown in Figure 5a, at a range of 10 km, the jammer started the game by selecting RSN, continued using RSN for a short duration, and then switched to VGPO. The switch resulted from a change in the values of the classification

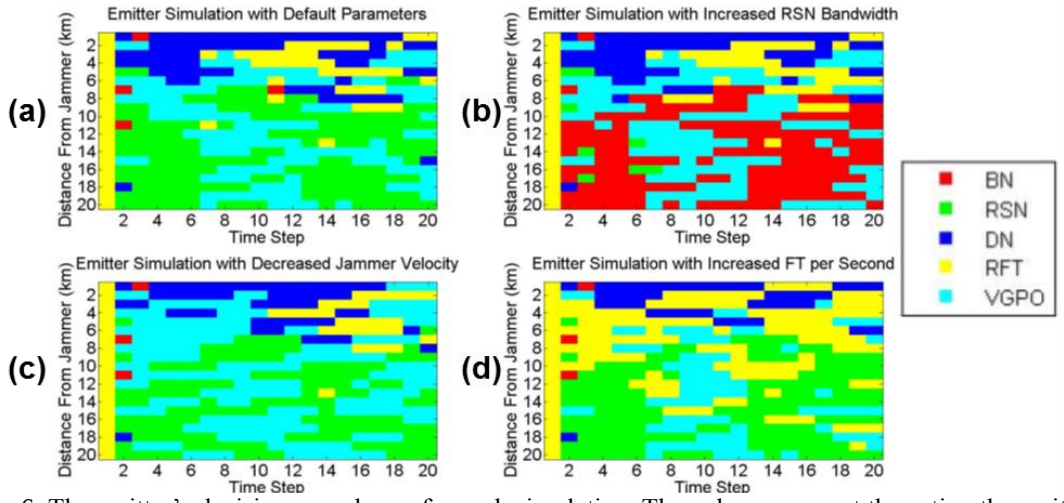

Figure 6. The emitter's decisions are shown for each simulation. The colors represent the action the emitter is countering, e.g. red represents Anti-BN.

matrix. The jammer became more confident that the emitter was in Anti-RSN mode, so it changed its action to DN. In Figure 6a, it is shown that the emitter was in fact using Anti-RSN, so the jammer was justified in switching. Likewise, once the emitter was able to determine that the jammer had switched to DN, it started to choose Anti-DN.

The first time step for each simulation shows which action is dominant at that range value. Since no measurements were received during the first time step, the classification matrix was uniform and did not contribute toward decision-making. Therefore, the action selected at the first time step was always the one that provides the most utility for the given range. The dominant attacks at extreme range values were DN for short-range attacks and RSN for long-range attacks because their utility functions scaled for short and long-range values, respectively. In the mid-range regions, neither DN nor RSN were as strong, so RFT and VGPO were more effective. Because the emitter's utility functions were designed to be entirely mode-dependent, the emitter acted as a purely responsive system. As a result, the emitter did not form its own regions where a particular action dominates. Instead, it mimicked the same regions as the jammer by responding to its actions. It always selected

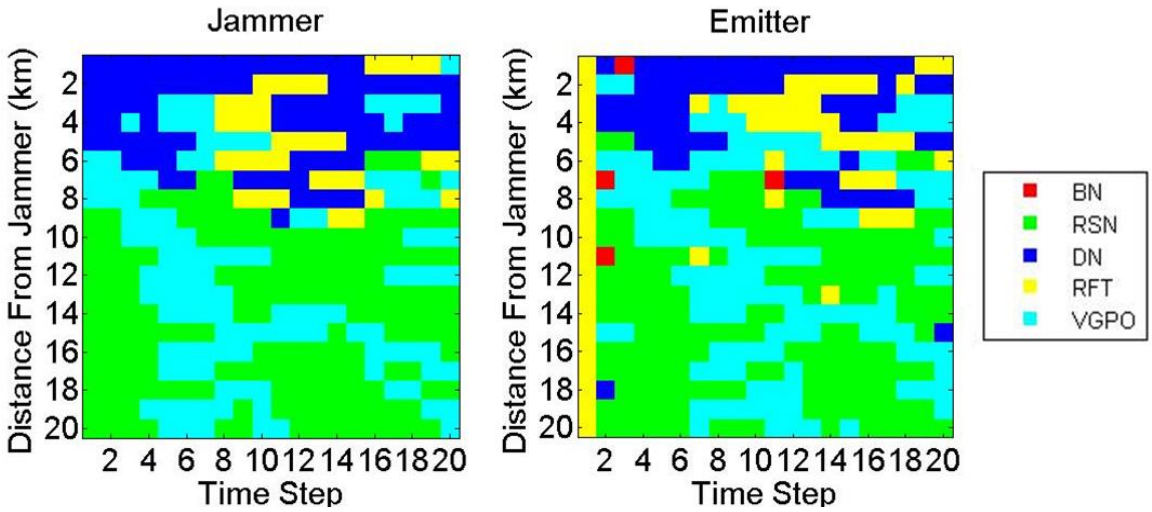

Figure 7. Results from the first simulation, which focuses on the effect of the distance between the jammer and the emitter on the agents' decisions, are shown for the jammer and the emitter. This simulation acts as a baseline for comparison with the other results.

RFT at time $t = 1$ because RFT provided the most utility for a uniform classification matrix.

The first simulation acted as a baseline for comparison against other simulations, and the results of this simulation are shown in Figure 7. Since the utility function for VGPO was designed to be range-independent, the first simulation showed the distances at which the range-based attacks became dominant over VGPO. While both BN and RSN attacks become more effective as the distance between the jammer and emitter increases, the utility functions for DN and RFT become less effective as the range increases. The simulation results reflected these claims and showed that the jammer started to choose RSN once the jammer and emitter were more than 9 km apart. BN was never chosen in the baseline simulation because the utility of BN was always strictly less than the utility of RSN. The emitter in this simulation behaved as expected by correctly protecting against DN in the short-range regions and against RSN in the long-range regions. Furthermore, the emitter was able to respond to the jammer's actions quickly without significant errors. The emitter's learning rate was responsible for this timely adaptation. While increasing the rate

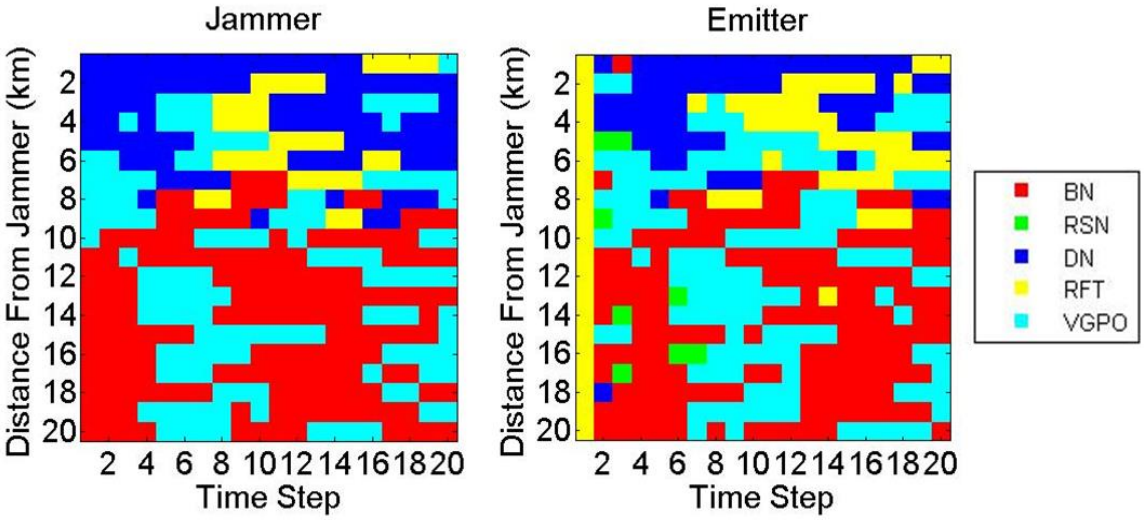

Figure 8. Results from the second simulation, which analyzes the effect of decreasing the utility of RSN, are shown for the jammer and the emitter.

would decrease response time, it would also increase the emitter's vulnerability to noisy measurements. For example, the emitter occasionally chose Anti-BN, even though the jammer never operated in that mode. The emitter received a measurement at every time step after the first for a total of 380 measurements. Since there was a 20% probability of receiving an incorrect measurement, the emitter was expected to receive around 76 incorrect measurements. In spite of this problem, the emitter still performed relatively well and did not select illogical actions frequently.

The second simulation showed the effect of increasing the bandwidth value used in the computation of the RSN utility; the results are shown in Figure 8. For this simulation, the bandwidth of the RSN attack was increased from 75 MHz to 150 MHz in order for the RSN bandwidth to be larger than the BN bandwidth. Since the JSR is inversely proportional to the bandwidth, the utility of RSN universally decreased, allowing other attacks to be selected in its place. The desired effect of this change was to observe whether or not BN would be selected instead. As expected, BN was the most commonly selected attack in place of RSN. However, since the bandwidth of BN in this simulation was still larger than the bandwidth of RSN used in the baseline simulation, the utility of BN in this simulation was lower than the utility of RSN in the baseline simulation. Because of this, BN did not entirely replace RSN. Since BN was less efficient than the baseline RSN, the jammer needed to be further away when performing BN to achieve the same utility. While RSN was dominant at 9 km, BN did not become dominant until the range was 11 km or higher. Once the jammer was sufficiently far away, BN became as dominant as RSN was in the baseline simulation. This fact is shown by Tables 3-5. In the baseline simulation (Figure 5a), the jammer chose RSN 36 times when the range was 10 km or closer and 167 times

Table 3. Lists the number of times the jammer selected each attack for each of the four simulations.

Number of Decisions	Simulation 1: Baseline	Simulation 2: Increased RSN Bandwidth	Simulation 3: Decreased Jammer Velocity	Simulation 4: Increased FTPS
BN	0	159	0	0
RSN	167	0	153	157
DN	83	87	72	60
VGPO	116	122	149	76
RFT	34	32	26	107

Table 4. Lists the number of times the jammer selected each attack for each of the four simulations when the range between the jammer and the emitter was greater than 10 km.

Number of Decisions	Simulation 1: Baseline	Simulation 2: Increased RSN Bandwidth	Simulation 3: Decreased Jammer Velocity	Simulation 4: Increased FTPS
BN	0	129	0	0
RSN	131	0	121	121
DN	0	0	0	0
VGPO	69	71	79	46
RFT	0	0	0	33

Table 5. Lists the number of times the jammer selected each attack for the four simulation when the range between the jammer and the emitter was less than 10 km.

Number of Decisions	Simulation 1: Baseline	Simulation 2: Increased RSN Bandwidth	Simulation 3: Decreased Jammer Velocity	Simulation 4: Increased FTPS
BN	0	30	0	0
RSN	36	0	32	36
DN	83	87	72	60
VGPO	47	51	70	30
RFT	34	32	26	74

overall. In the second simulation (Figure 5b), the jammer chose BN 159 times yet only 30 times (20% less often) when the range was less than 10 km. Since both attacks were chosen around the same number of times in their respective simulations when the range was above 10 km (131 and 129 times respectively), the drop in the number of close-range decisions supports the conclusion that the decision region was shifted toward higher range values. In this simulation, the emitter behaved as expected: its decisions changed from Anti-RSN to Anti-BN in response to the jammer. The emitter still behaved accurately in the short-range regions by defending itself from DN properly and, when the jammer switched to VGPO or RFT, by quickly switching to the correct defense.

The third simulation, in which the speeds of both the jammer and the false track were reduced from 300 m/s to 250 m/s, is illustrated in Figure 9. This change increased the amount of time required for the jammer and the false track to separate from each other, allowing the jammer to deceive the emitter for a longer duration. As expected, this change increased the number of times that VGPO was selected by the jammer. In the baseline simulation, VGPO was selected 116 out of a maximum of 400 times. In the simulation with decreased jammer velocity, VGPO was selected 149 times for an increase of 28 percent

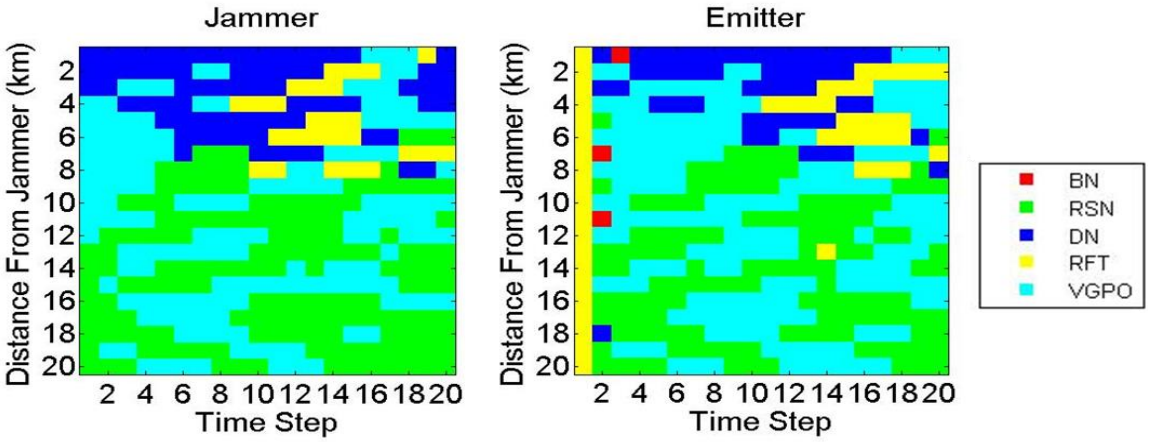

Figure 9. Results from the third simulation, which focuses on increasing the utility of VGPO attacks by decreasing the Jammer's velocity, are shown for the jammer and the emitter.

over baseline. Throughout the simulation, the emitter behaved as expected by correctly responding to the jammer's changing strategy. As a result, it ended up creating decision regions similar to the jammer. Furthermore, the emitter appeared to be trailing just behind the jammer in many instances.

The final simulation, shown in Figure 10, focused on increasing the number of false targets the jammer can generate in one second, which directly increased the utility for RFT by increasing θ_{RFT} (the main parameter of the RFT utility function, given by (13)). For this simulation, the number of false targets generated was doubled, and the number of RFT decisions more than tripled from 34 to 107. This result was surprising because the utility function for RFT is exponential for θ_{RFT} greater than 1 (see equation (14)), and the value of θ_{RFT} must therefore increase dramatically in order to raise the utility by even a small amount. Since all of the noise-based attacks scale significantly better with range than RFT scales with θ_{RFT}, it was hypothesized that RFT would be unable to overpower the noise-based attacks at extreme range value and, therefore, that the increase in the number of RFT

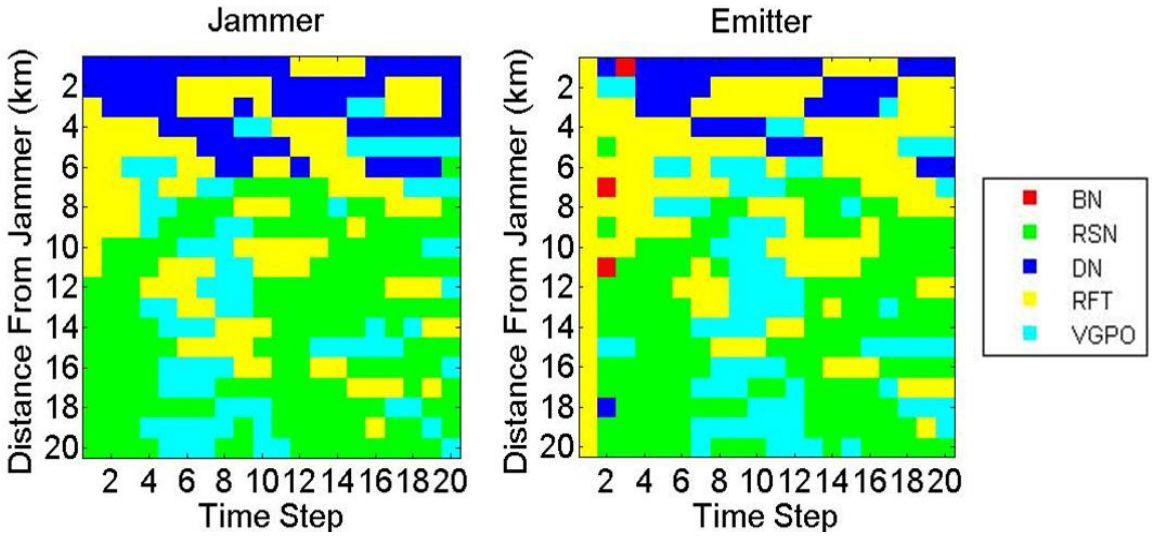

Figure 10. Results from the final simulation, which analyzes the effects of increasing RFT's utility, are shown for the jammer and the emitter.

decisions would be minimal. However, RFT was very successful for medium and close range values and was also occasionally chosen at longer ranges.

The fourth simulation chose VGPO only 76 times, which is a drop of 34 percent from baseline, and chose DN only 60 times (down from 83), which is a decrease of 28 percent. The number of times RSN was chosen, however, only decreased by six percent from 167 down to 157. These results occurred because of the slight range dependence inherent in the RFT utility function. As the range between the two agents increased, the jammer had less time to generate false targets because of the increased time of arrival. In other words, because the jammer was further away from the emitter, not all of the false targets generated had time to reach the emitter before the end of the time step. As a result, the RFT utility function was less effective at longer ranges, which was an unexpected consequence of the utility function's design.

CHAPTER 5

Conclusion and Future Work

This thesis introduces a probabilistic framework for rational decision-making in wireless environments and presents an example application in the field of electronic warfare. By design, the probabilistic framework greedily and myopically maximizes the utility gained from choosing an action, and this objective is realized through the cooperation of three modules: the adversary identification module which utilizes a knowledge base of data to infer the behavior of the other agents, the utility calculator module which evaluates the benefit of each action according to predefined utility functions and observables, and the optimization module which ranks the actions available to the agent. The versatility of these modules allows general application to a variety of wireless electronic support applications.

The example problem provided in Section 3.0 demonstrates an implementation of the general framework. Under simplifying assumptions, a contest between a jammer and an emitter is simulated under a variety of conditions, where the goal of each agent is to derive rational decisions from simple models and observed data. Both agents utilize the framework for this purpose in the simulations. The results from the simulations demonstrate the behavior of the agents to be both rational and in accordance with expectations, and while the agents develop clear decision boundaries where one action dominates all others, the results show that the agents are capable adaptation in a reasonable amount of time.

It is worth mentioning that the utility functions used in the simulation are designed primarily for the purposes of demonstration and experimentation and do not necessarily represent the best way to evaluate an action. The utility functions are intentionally designed to be different from each other in order to provide insight into how different styles of utility functions operate, interact, and influence the decision process. While the utility functions are not completely realistic, they are still an effective demonstration of how the generic framework can be used to provide rational decision support.

Further work could focus on improving upon the weaknesses of the framework. The current framework is limited to two-player applications. For adversaries, this restriction is easily addressed by abstracting the classification matrix into three dimensions (as shown in Figure 3) to account for multiple opponents. The optimization module would then seek to choose the action that maximizes the weighted sum of the expected utility against each opponent. Generalizing the framework to allow for cooperation between friendly agents is more difficult. The main areas of focus could include adapting the adversary identification module to incorporate shared information and shifting toward a centralized optimizer that could distribute the combined capabilities and resources of the allied agents optimally. This extension would be a promising area for further research. Another limitation is the myopic nature of the framework; i.e. the current framework only maximizes the current utility. From a game-theoretic perspective, planning ahead in order to account for an opponent's strategy is often advantageous; a greedy strategy is not necessarily optimal over the course of several time steps. Through clever use of the models stored in the knowledge base, prediction of the future actions of the opponent would be possible and would allow the agent to plan accordingly.

REFERENCES

[1] D. Poole and A. Mackworth, *Artificial Intelligence: Foundations of Computational Agents*. Princeton, NJ. Princeton University Press, 2010.

[2] D. Wonderley, T. Selee, and V. Chakravarthy, “Game theoretic decision support framework for electronic warfare applications,” in *2016 IEEE Radar Conference*, Philadelphia, PA, 2016, pp. 1-5.

[3] J. V. Neumann and O. Morgenstern, *Theory of Games and Economic Behavior*. Princeton, NJ. Princeton University Press, 1944.

[4] A. Partizian, “Electronic Protection,” in *Principles of Modern Radar: Advanced Techniques*, W.L. Melvin and J.A. Scheer, Ed. Edison: Scitech Publishing, 2013, ch. 12, pp. 529-553.

[5] A. Farina, “Electronic Counter-Countermeasures,” in *Radar Handbook*, M. Skolnik, 3rd ed. New York: McGraw-Hill, 2008, pp. 24.1-24.59.

[6] G. E. Smith et al., “Radar classification evaluation,” in *Proceedings of the 2008 IEEE Radar Conference*, Rome, Italy, May 2008.

[7] P. D. Gader et al., "Landmine detection with ground penetrating radar using hidden Markov models," in *IEEE Transactions on Geoscience and Remote Sensing*, vol. 39, no. 6, pp. 1231-1244, Jun 2001.

[8] A. Kale et al., "Identification of humans using gait," *in IEEE Transactions on Image Processing*, vol. 13, no. 9, pp. 1163-1173, Sept. 2004.

[9] P. S. Aleksic and A. K. Katsaggelos, "Automatic facial expression recognition using facial animation parameters and multistream HMMs," in *IEEE Transactions on Information Forensics and Security*, vol. 1, no. 1, pp. 3-11, Mar. 2006.

[10] T. D. Bufler and R. M. Narayanan, "Radar classification of indoor targets using support vector machines," in *IET Radar, Sonar & Navigation*, vol. 10, no. 8, pp. 1468-1476, Oct. 2016.

[11] G. Scutari et al., "Convex optimization, game theory, and variational inequality theory," in *IEEE Signal Processing Magazine*, vol. 27, no. 3, pp. 35-49, May 2010.

[12] P. Reyes, “Logistics networks: A game theory application for solving the transshipment problem,” in *Applied Mathematics and Computation*, vol. 2, pp. 1419-1431, Sept. 2005.

[13] Y. Meng, “Multi-robot searching using game theory based approach,” in *International Journal of Advanced Robotic Systems*, Vol. 5, Nov. 2008.

[14] A. W. Moore and C. G. Atkeson, “The Parti-game algorithm for variable resolution reinforcement learning in multidimensional state-spaces,” in *Machine Learning*, Vol. 21, pp. 199-233, 1995.

[15] K. Madani, “Hydropower licensing and climate change: Insights from cooperative game theory,” in *Advances in Water Resources*, Vol 34, pp. 174-183, Feb. 2011.

[16] M. J. E. Skardi et al., “Simulation-optimization model for non-point source pollution management in watersheds: Application of cooperative game theory,” in *KSCE Journal of Civil Engineering*, Vol 17, pp. 1232-1240, Sept. 2013.

[17] C. K. Doshi et al., “Game theoretic modeling of grey hole attacks in wireless ad hoc networks,” in *Proceedings of the International Conference on Signal, Networks, Computing, and Systems*, Vol 1, pp. 217-226, 2016.

[18] J. Zuh et al., “A game-theoretic power control mechanism based on HMM in cognitive wireless sensor network with imperfect information,” in *Neurocomputing*, vol. 220, pp. 76-83, Jan. 2017.

[19] B. Béjar et. al, “Cooperative localisation in wireless sensor networks using coalitional game theory,” in *2010 18th European Signal Processing Conference*, Aalborg, pp. 1459-1463, 2010.

[20] F. Picard, “An Introduction to Mixture Models,” Stat. for Syst. Biology Group, Paris, France, Rep. 7, Mar. 2007.

[21] C. J. C. Burges, “A Tutorial on Support Vector Machines for Pattern Recognition,” in *Data Mining and Knowledge Discovery*, vol. 2, 1998, pp. 121-167.

[22] Y. Bar-Shalom et al., "The probabilistic data association filter," in *IEEE Control Systems*, vol. 29, no. 6, pp. 82-100, Dec. 2009.

[23] N. Petrov, I. Jordanov, and J. Roe, “Identification of Radar Signals using Neural Network Classifier with Low-Discrepancy Optimisation,” in *IEEE Congress on Evolutionary Computation*, Cancun, 2013, pp. 2658-2664.

[24] J. Liu et al., “The Minimum Description Length Criterion Applied to Emitter Number Detection and Pulse Classification,” in *Statistical Signal and Array Processing*, Portland, OR, 1998, pp. 172-175.